

---

# Operating System Support for Process Confinement

---

José R. Herrero, David Benlliure

[josepr@ac.upc.es](mailto:josepr@ac.upc.es)

Computer Architecture Department  
Facultat d'Informàtica de Barcelona  
Universitat Politècnica de Catalunya  
Barcelona, Spain



SAM'03, Las Vegas, 06/24/2003

1

# Talk Outline

---

- Introduction: goal, concepts, motivation, ...
- Understanding the issue
- OS support
- Previous work
- Our contribution
- Conclusion
- Questions



# Goal

---

- Provide a user level tool for process confinement



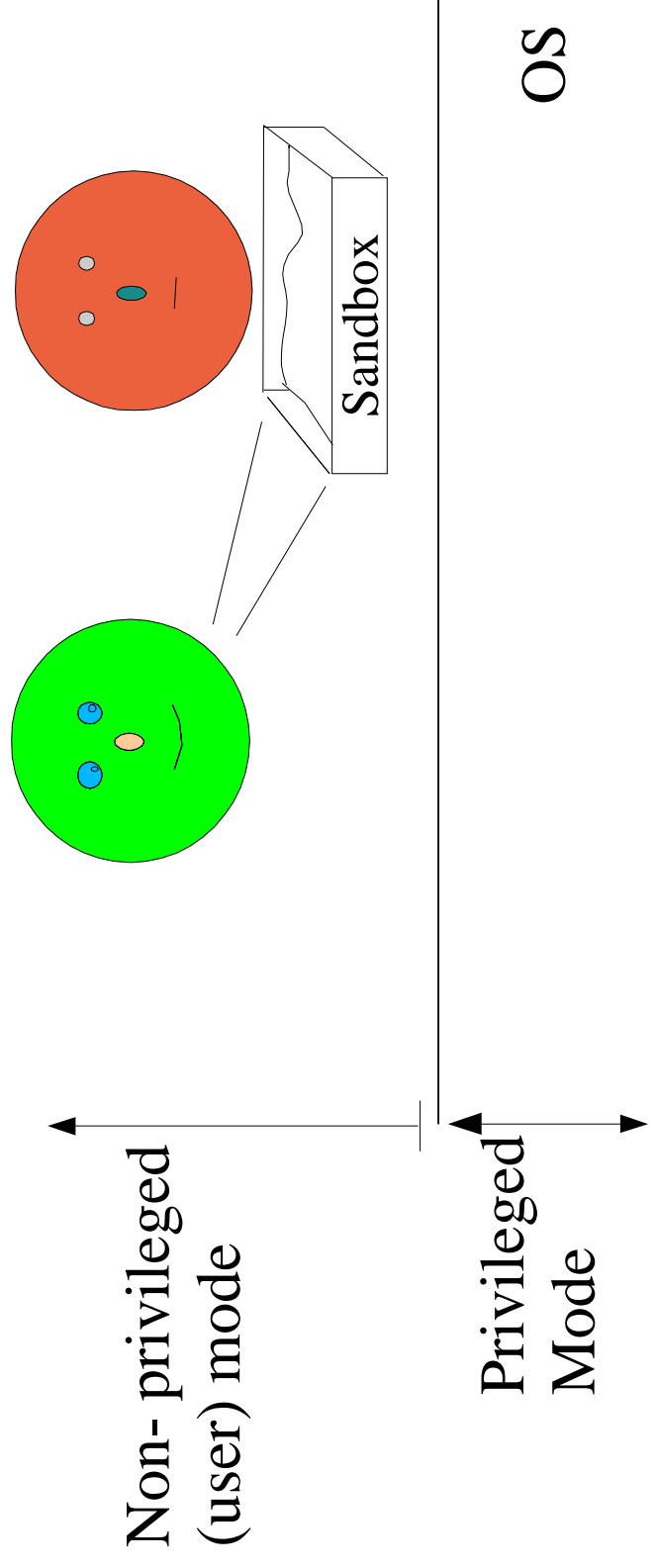
# Concepts

---

- Process confinement
    - Ability to limit what a process can do
  - User level tool
    - A program than can be run without special privileges
- ⇒ Can be run by any user in the system

# Idea

---



# Motivation

---

- Threats to the system can come from *inside*
- Authorized users executing *external* **untrusted** software
- Downloaded from the web
- Received as E-mail attachments
- ...

Trojan Horses



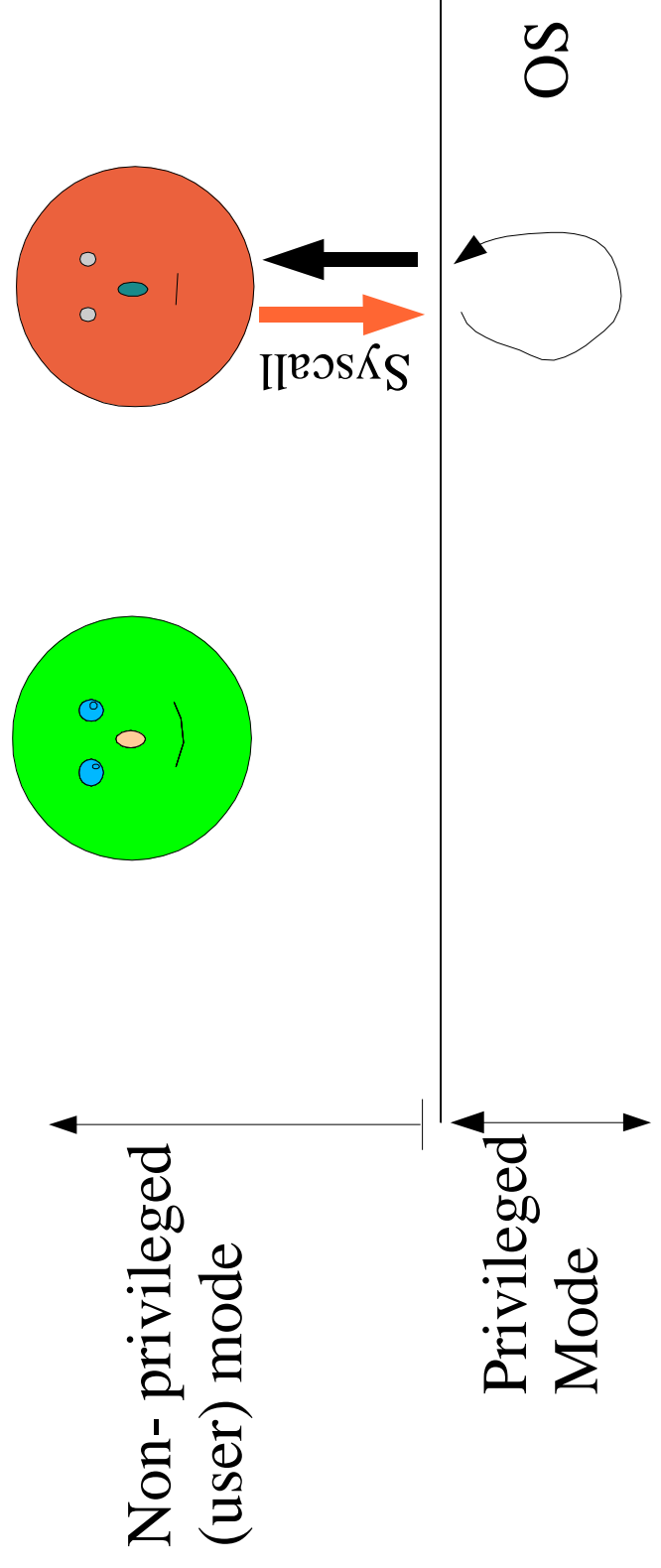
# Requirements

---

- To be able to build a user level tool which controls and limits what a process does we need ...
  - Time
  - Patience
  - Skills :o)
  - Operating System support



# How do processes get work done?

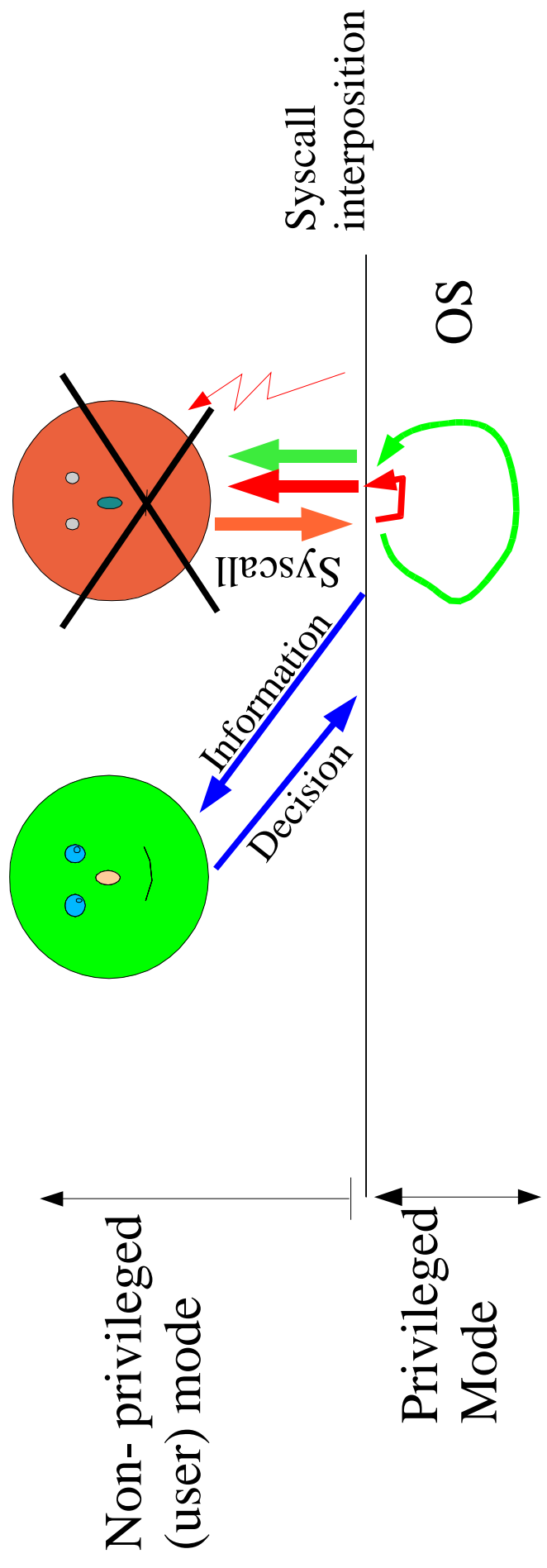


Hardware



# What can we do?

---



Hardware

# OS support for syscall interposition

- UNIX
  - Sys V
    - /proc
      - read/write
      - ioctl
    - ptrace



# OS support for syscall interposition

- UNIX
  - Linux
    - /proc
      - gives information about processes
      - does NOT allow for any control
    - ptrace
      - does not meet all the requirements
        - × Ptrace\_Kill kills process after the system call is serviced
        - × trace flags are not inherited (children could run away from control)

# Related work

---

- *Janus*
  - Solaris
    - Uses /proc
  - Linux
    - They create a whole kernel module  
( $\approx$  3000 lines of code)

# Our contribution

---

- UNIX
    - Linux
      - /proc (future contrib to linux? + paper?)
      - Currently we have a prototype working with basic control functionalities
        - ✓ stop/resume/kill
        - ✓ offered via ioctl
    - ptrace (this paper)
      - Extended to allow for user level process confinement
- (≈ 50 lines of code)



# Our contribution

---

- Changes to the Linux kernel
  - *ptrace()*
    - 2 extra commands
      - *PTRACE\_TRACEUS*
      - *PTRACE\_DESTROY*
  - *task\_struct*
    - 2 new flags
      - *PF\_TRACEINHERIT*
      - *PF\_DESTROY*
  - *fork()*
  - *syscall\_trace()*



# Conclusions

---

- We needed to extend the Linux kernel to provide
  - Denial of system call
  - Inheritance of trace flags across *fork()* calls.
- We could implement a user level tool to allow for process confinement
- All operating systems should provide support for process confinement

# Questions

---

- What about *Windows* operating systems?
  - Do you have detailed information about system calls?
    - What *MS* calls the *Native NT API* but does not document publicly
    - Not the *Win32 API*!

